UV Sensor v1.0-ML8511 SKU: SEN0175

UV Sensor v1.0-ML8511

Contents

Introduction

The ML8511 is a UV sensor, which is suitable for acquiring UV intensity indoors or outdoors. iT is equipped with an internal amplifier, which converts photo-current to voltage depending on the UV intensity. This unique feature offers an easy interface to external circuits such as ADC. In the power down mode, typical standby current is 0.1A, thus enabling a longer battery life.

This sensor detects 280-390nm light most effectively. This is categorized as part of the UVB (burning rays) spectrum and most of the UVA (tanning rays) spectrum. It outputs a analog voltage that is linearly related to the measured UV intensity (mW/cm2). If your microcontroller can do an analog to voltage conversion then you can detect the level of UV.

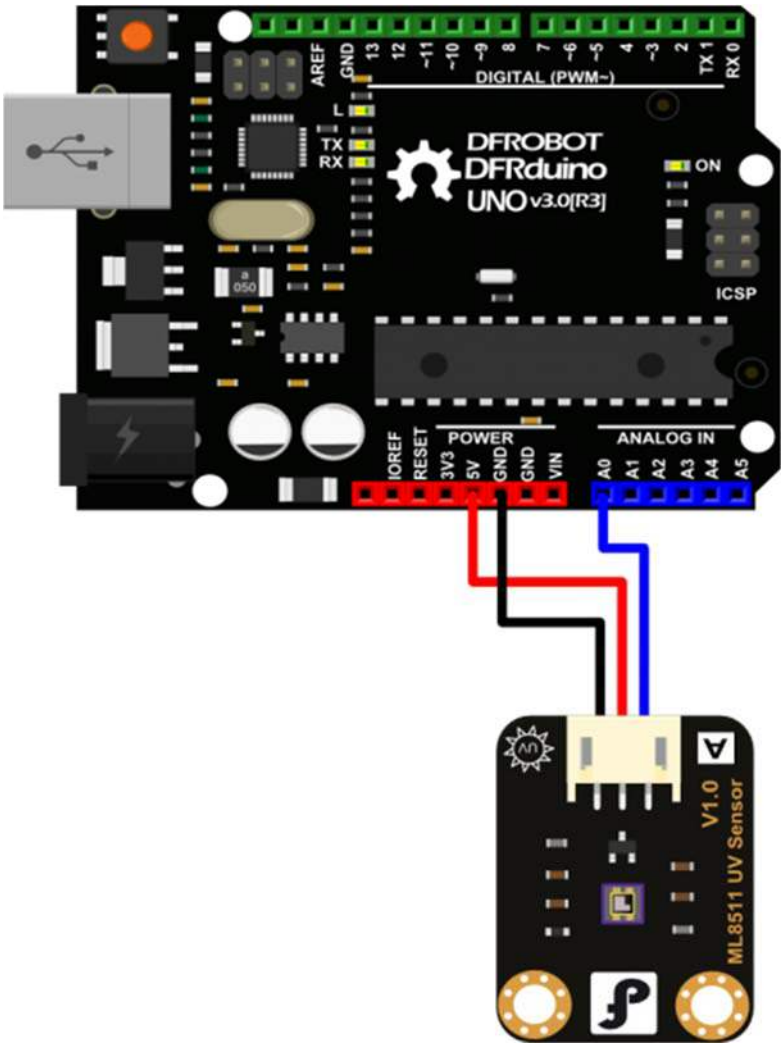
Applications

- Weather Station
- UV Index Monitoring
- DIY UV electronic project,etc...

Specification

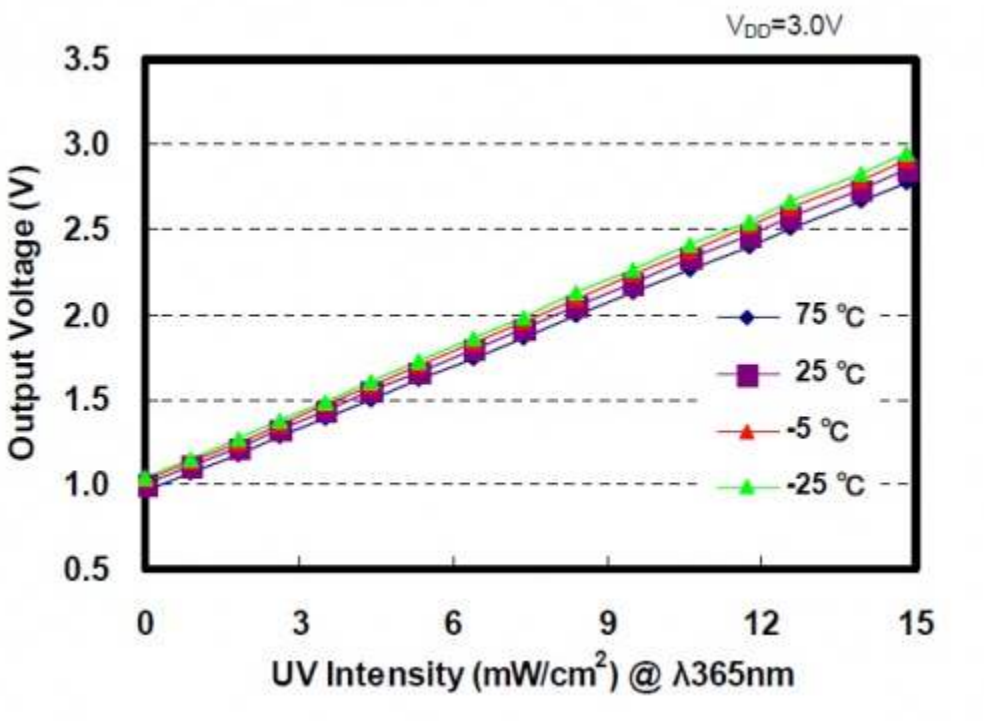
- Supply Voltage: DC 5V
- Operating Temperature: -20~70°C
- Sensitivity Region：UV-A and UV-B
- Sensitivity Wave Length:280-390nm
- Module Size: 30 x 22mm

Connection Diagram



ML8511 UV Sensor Diagram

The ML8511 intensity graph



The ML8511 intensity graph

Mapping the outputVoltage to intensity is straight forward. No UV light starts at 1V with a maximum of 15mW/cm2 at around 2.8V. Arduino has a built-in map() function, but map() does not work for floats. Thanks to users on the Arduino forum, we have a simple mapFloat() function:

```
//The Arduino Map function but for floats
//From: http://forum.arduino.cc/index.php?topic=3922.0
float mapfloat(float x, float in_min, float in_max, float out_min, float out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

The following line converts the voltage read from the sensor to mW/cm2 intensity:

```
float uvIntensity = mapfloat(outputVoltage, 0.99, 2.8, 0.0, 15.0); //Convert
the voltage to a UV intensity level
```

Sample Code

```
/*************************************************

 * UV Sensor v1.0-ML8511

 * <http://www.dfrobot.com/index.php?route=product/product&product_id=1195&se
arch=uv&description=true>

 *************************************************

 * This example reads UV intensity from UV Sensor v1.0-ML8511.

 *

 * Created 2014-9-23

 * By Phoebe <phoebe.wang@dfrobot.com>

 * Modified 2014-9-23

 * By Phoebe phoebe.wang@dfrobot.com>

 *

 * GNU Lesser General Public License.

 * See <http://www.gnu.org/licenses/> for details.

 * All above must be included in any redistribution

 *************************************************/


/***********Notice and Trouble shooting***************

 * 1.Connect ML8511 UV Sensor to Arduino A0

   <http://www.dfrobot.com/wiki/index.php/File:SEN0175_Diagram.png>

 * 2.This code is tested on Arduino Uno, Leonardo, Mega boards.

 *************************************************/



int ReadUVintensityPin = A0; //Output from the sensor


void setup()

{

  pinMode(ReadUVintensityPin, INPUT);
```

```
  Serial.begin(9600); //open serial port, set the baud rate to 9600 bps
  Serial.println("Starting up...");
}

void loop()
{
  int uvLevel = averageAnalogRead(ReadUVintensityPin);

  float outputVoltage = 5.0 * uvLevel/1024;
  float uvIntensity = mapfloat(outputVoltage, 0.99, 2.9, 0.0, 15.0);

  Serial.print("UVAnalogOutput: ");
  Serial.print(uvLevel);

  Serial.print(" OutputVoltage: ");
  Serial.print(outputVoltage);

  Serial.print(" UV Intensity: ");
  Serial.print(uvIntensity);
  Serial.print(" mW/cm^2");

  Serial.println();
  delay(100);
}

//Takes an average of readings on a given pin
//Returns the average
int averageAnalogRead(int pinToRead)
{
  byte numberOfReadings = 8;
  unsigned int runningValue = 0;

  for(int x = 0 ; x < numberOfReadings ; x++)
```

```
    runningValue += analogRead(pinToRead);
  runningValue /= numberOfReadings;


  return(runningValue);


}


//The Arduino Map function but for floats
//From: http://forum.arduino.cc/index.php?topic=3922.0
float mapfloat(float x, float in_min, float in_max, float out_min, float out_
max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```